UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/981,453	10/18/2001	Jens B. Junkermann	10022/093	6887

33391 7590 05/31/2006

BRINKS HOFER GILSON & LIONE
ONE INDIANA SQUARE, SUITE 1600
INDIANAPOLIS, IN 46204

EXAMINER	
STEVENS, ROBERT	
ART UNIT	PAPER NUMBER
2176	

DATE MAILED: 05/31/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/981,453	JUNKERMANN, JENS B.	
	Examiner	Art Unit	
	Robert Stevens	2176	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 March 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *21,22,24-30,32,33,41-77 and 79* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *21,22,24-30,32,33,41-77 and 79* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to communications: amendment filed 3/14/2006.

2. This action is **FINAL**.

3. The Office withdraws the previous rejections of claims 21-22, 24-30, 32-33, 41-50, 64-77 and 79 under 35 U.S.C. § 103(a) as being unpatentable over Burkett in view of Allen, in light of the amendment.

4. The Office maintains the previous rejections of claims 51-53 under 35 U.S.C. § 103(a) as being unpatentable over Burkett in view of Allen, in light of the amendment.

5. The Office asserts new rejections of claims 21-22, 24-30, 32-33, 41-50, 64-77 and 79 under 35 U.S.C. § 103(a) as being unpatentable over Burkett in view of Lection, in light of the amendment.

6. Claims 21-22, 24-30, 32-33, 41-77 and 79 are pending. Claims 21, 41, 51 and 64 are independent. Claims 1-20, 23, 31, 34-40 and 78 have been canceled.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. **Claims 21-22, 24-30, 32-33, 41-50, 64-77 and 79 are rejected under 35 U.S.C. 103(a)** as being unpatentable over Burkett et al. (US Patent No. 6,635,089, filed Jan. 13, 1999 and issued Oct. 21, 2003, hereafter referred to as "Burkett") in view of Lection et al (US Patent No. 6,418,446, filed Mar. 1, 1999 and issued Jul. 9, 2002, hereafter referred to as "Lection").

Independent claim 21 states:

A method of operating a business services application for retrieving data with delivery technologies, the method comprising:

developing custom application code in a subclass of a BusinessService class, the custom application code responsive to a request for data initiated by the delivery technologies;

translating the request to a first document object model document with an ApiService class;

during the translation limiting the data structure of the first document object model document to representation as an input message with ;plurality of fields, wherein units of data included in each of the fields is limited to a data type that is pre-specified in the business services application;

executing the custom application code to retrieve data based on the first document object model document;

reading data into a second document object model document with the ApiService class;

while the data is read in, self limiting the data structure of the second document object model document to representation as an output message with a

plurality of fields wherein units of data included in each of the fields is limited to a data type that is pre-specified in the business services application; and translating the second document object model document with, the ApiService class based on the delivery technology.

Burkett teaches the well-known use of application programs for implementing functionality in software in col. 3 lines 28-35, discussing the execution of application programs, it being implicit that if one executed an application program that the program has been developed. Burkett further teaches implementations in an e-commerce environment in col. 15 lines 18-46, discussing electronic payments and bank account processing. Burkett teaches the well-known concept of translating between documents in col. 3 lines 17-35, discussing API-based processing and the updating of documents. See also See also col. 4 lines 21-36. It is inherent/implicit in DOM processing taught by Burkett that Dom "fields" (i.e., nodes) are constructed from XML data (i.e., tags), as discussed in col. 2 lines 44-52. It is further well-known that DOMs were employed for the purpose of translating among XML documents, as discussed in col. 3 lines 17-27 of Burkett, it being implicit that a second data format was achieved via use of the DOM document model.

However, Burkett does not explicitly disclose pre-defined data types. Lection, though, teaches pre-defined data types in col. 8 lines 35-65, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record

formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

Regarding dependent claims 22, 24-30 and 32-33, Burkett teaches the well-known use of XML, HTML and format changes in col. 1 lines 16-35 and 59-65, discussing XML and format transformation, the use of XSL being well-known in the art of format transformations in an XML environment.

However, Burkett does not explicitly disclose various specific datatypes. Lection, though, teaches the use of several specific datatypes associated with XML tags in col. 8 lines 35-53, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags. Lection further discloses populating attribute nodes with attributes read in / formatted in col. 9 lines 14-24 discussing the association of keyword identifiers with values.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the

same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

Independent claim 41 states:

A system for leveraging extensible markup language technology to provide an interface between a back-end systems layer and a front-end systems layer, the system comprising:

a server computer;

an ApiService class operable within the server computer to direct the translation of a request to an input message that includes a plurality of fields;

a document object model class operable within the server computer to represent the input message as a document object model document;

a Message class and a Field class operable within the server computer as wrapper of the document object model class to restrict manipulation and standardize the content of the document object model document;

a MESSAGEDEFINITION class operable in the server, wherein the MESSAGEDEFINITION class includes a listing of pre-specified fields each of which describe a corresponding pre-specified data type, and wherein the Message class and the Field class are further operable within the server during translation to limit a format of corresponding fields included in the input message to a predetermined data structure based on the described corresponding pre-specified data type; and

a BusinessService class operable within the server computer to direct the execution of custom application code as a function of the input message, wherein the custom application code includes a pre-specified data type to limit the format of those fields included in the input message that do not correspond to the listing of pre-specified fields.

Burkett teaches the well-known use of servers in col. 6 lines 8-20, discussing a network processing environment utilizing an IBM Enterprise Systems Architecture. Burkett teaches the well-known use of application programs for implementing functionality in software in col. 3 lines 28-35, discussing the execution of application programs, it being implicit that if one executed an application program that the program has been developed. Burkett further teaches implementations in an e-commerce environment in col. 15 lines 18-46, discussing electronic

payments and bank account processing. Burkett teaches the well-known concept of translating between documents in col. 3 lines 17-35, discussing API-based processing and the updating of documents. See also See also col. 4 lines 21-36. It is inherent/implicit in DOM processing taught by Burkett that Dom "fields" (i.e., nodes) are constructed from XML data (i.e., tags), as discussed in col. 2 lines 44-52. It is further well-known that DOMs were employed for the purpose of translating among XML documents, as discussed in col. 3 lines 17-27 of Burkett, it being implicit that a second data format was achieved via use of the DOM document model. It is further well-known that DOM, which expands to Document Object Model, processing utilized object-oriented techniques (i.e., class objects/variables). The specific class name was merely an obvious variant to one of ordinary skill in the art at the time of the invention.

However, Burkett does not explicitly disclose pre-defined data types. Lection, though, teaches pre-defined data types in col. 8 lines 35-65, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

Regarding dependent claims 42-50, Burkett teaches that DOM-processing is tree based in col. 1 lines 35-58, discussing tree-oriented abstraction of a document, it being well-known that trees are comprised of parent and children nodes, and further discussing that DOM-processing was employed to implement a translation mechanism between source and target documents. Burkett further teaches implementations in an e-commerce environment in col. 15 lines 18-46, discussing electronic payments and bank account processing.

However, Burkett does not explicitly disclose various specific datatypes and class variable names. Lection, though, teaches the use of several specific datatypes associated with XML tags in col. 8 lines 35-53, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags. Lection further discloses populating attribute nodes with attributes read in / formatted in col. 9 lines 14-24 discussing the association of keyword identifiers with values.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

Independent claim 64 states:

An e-commerce architecture for providing a framework to interface delivery technologies with data, the e-commerce architecture comprising:

a server computer operable to execute instructions to convert a request to a first document object model document in an extensible markup language, the first document object model document comprising a plurality of request parameters extracted from the request;

the server computer operable to execute instructions to restrict the conversion to the first document object model document to based on a listing of data types that arc pre-specified for the request parameters, wherein the data types limit the data structure of a plurality of fields included in the first document. object model document to a predetermined data structure specified by the data types;

the server computer operable to execute instructions to retrieve data responsive to the request and convert the data to a second document object model document in the extensible markup language based on the request parameters; and

the server computer operable to execute instructions to restrict the conversion of the data to the second document object model document to limit the data structure of a plurality of fields included in the second document object model document to a predetermined data structure specified by the data types.

Burkett teaches the well-known use of servers in col. 6 lines 8-20, discussing a network processing environment utilizing an IBM Enterprise Systems Architecture. Burkett teaches the well-known use of application programs for implementing functionality in software in col. 3 lines 28-35, discussing the execution of application programs, it being implicit that if one executed an application program that the program has been developed. Burkett further teaches implementations in an e-commerce environment in col. 15 lines 18-46, discussing electronic payments and bank account processing. Burkett teaches the well-known concept of translating between documents in col. 3 lines 17-35, discussing API-based processing and the updating of documents. See also See also col. 4 lines 21-36. It is inherent/implicit in DOM processing taught by Burkett that Dom "fields" (i.e., nodes) are constructed from XML data (i.e., tags), as

discussed in col. 2 lines 44-52. It is further well-known that DOMs were employed for the purpose of translating among XML documents, as discussed in col. 3 lines 17-27 of Burkett, it being implicit that a second data format was achieved via use of the DOM document model. It is further well-known that DOM, which expands to Document Object Model, processing utilized object-oriented techniques (i.e., class objects/variables). The specific class name was merely an obvious variant to one of ordinary skill in the art at the time of the invention.

However, Burkett does not explicitly disclose pre-defined data types. Lection, though, teaches pre-defined data types in col. 8 lines 35-65, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

Regarding dependent claims 65-70, Burkett teaches that DOM-processing is tree based in col. 1 lines 35-58, discussing tree-oriented abstraction of a document, it being well-known that trees are comprised of parent and children nodes, and further discussing that DOM-processing was employed to implement a translation mechanism between source and target documents. Burkett further teaches implementations in an e-commerce environment in col. 15 lines 18-46, discussing electronic payments and bank account processing.

However, Burkett does not explicitly disclose various specific datatypes and class variable names. Lection, though, teaches the use of several specific datatypes associated with XML tags in col. 8 lines 35-53, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags. Lection further discloses populating attribute nodes with attributes read in / formatted in col. 9 lines 14-24 discussing the association of keyword identifiers with values.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

Regarding dependent claims 71-77 and 79, Burkett teaches that DOM-processing is tree based in col. 1 lines 35-58, discussing tree-oriented abstraction of a document, it being well-known that trees are comprised of parent and children nodes, and further discussing that DOM-processing was employed to implement a translation mechanism between source and target documents. Burkett further teaches implementations in an e-commerce environment in col. 15 lines 18-46, discussing electronic payments and bank account processing.

However, Burkett does not explicitly disclose various specific datatypes and class variable names. Lection, though, teaches the use of several specific datatypes associated with XML tags in col. 8 lines 35-53, discussing keyword tags, such as <STRING>, <NUMBER> and <GROUP>, which limit the associated data to the format corresponding to those tags. Lection further discloses populating attribute nodes with attributes read in / formatted in col. 9 lines 14-24 discussing the association of keyword identifiers with values.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Lection for the benefit of Burkett, because to do so would have allowed a system designer to provide a technique whereby data, having dynamically variable record formats, can be easily and efficiently accommodated by program code processing that data without requiring modification of the processing code each time the underlying data format changes, as taught by Lection in col. 2 lines 59-65. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion, and the inventive entities include overlapping inventors.

9. **Claims 51-63 are rejected under 35 U.S.C. 103(a)** as being unpatentable over Burkett et al. (US Patent No. 6,635,089, filed Jan. 13, 1999 and issued Oct. 21, 2003, hereafter referred to as "Burkett") in view of Allen (US Patent No. 6,658,625, filed Apr. 14, 1999 and issued Dec. 2, 2003, hereafter referred to as "Allen").

Regarding independent claim 51, Burkett discloses:

> *A method of leveraging extensible markup language technology to interface a front-end systems layer and a back-end systems layer* (Fig. 2), *the method comprising:*
> *receiving one of a plurality of predetermined requests initiated with any one of a plurality of delivery technologies;* (Abstract and Fig. 2 in context of col. 3 lines 17-26)
> *converting the request to a plurality of fields based on request parameters included in the request;* (Abstract in context of col. 3 lines 17-26)
> *limiting a datatype of data included in the fields to a predefined group of datatypes;* (Fig. 10 and col. 16 line 60 – col. 17 line 24)
> ... ; *and*
> ... .

However, Burkett does not explicitly disclose:

> ...
> ... ;
> ... ;
> ... ;
> *extracting the request parameters based on the datatype; and*
> *accessing data responsive to the request based on the extracted request parameters.*

Allen, though, discloses:

> ...
> ... ;
> ... ;

... ;

extracting the request parameters based on the datatype; (Abstract and col. 13 lines 27-67) *and*

accessing data responsive to the request based on the extracted request parameters. (Abstract and col. 13 lines 27-67)

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Allen for the benefit of Burkett, because to do so would have allowed a system designer to implement a data converter that uses the data description to determine how to convert the data, as taught by Allen in the Abstract. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion.

Regarding claim 52, which is dependent upon claim 51, Burkett further discloses:

wherein the datatype of data included in the fields is predefined by the request. (col. 4 lines 21-64)

Regarding claim 53, which is dependent upon claim 51,

Burkett does not explicitly disclose:

wherein the datatype of data included in the fileds is loaded from a static declaration of the datatype included in a MESSAGEDEFINITION class.

Allen, though, discloses:

wherein the datatype of data included in the fileds is loaded from a static declaration of the datatype included in a MESSAGEDEFINITION class. (Abstract and Fig. 2, 3A, 3B)

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply the teachings of Allen for the benefit of Burkett, because to do so would have allowed a system designer to implement a data converter that uses the data description to determine how to convert the data, as taught by Allen in the Abstract. These references were all applicable to the same field of endeavor, i.e., use of tree processing techniques for data conversion.

Regarding claim 54, which is dependent upon claim 51, Burkett further discloses:

> *wherein converting the request comprises translating the request to XML structure that is limited to the predefined group of datatypes.* (col. 3 lines 59-67 and col. 15 lines 4-67)

Regarding claim 55, which is dependent upon claim 51, Burkett further discloses:

> *converting the data responsive to the request into a plurality of fields with a data type that is limited to the predefined group of datatypes based on the request parameters, and translating the fields into a format indicated by the request to be compatible with the one of the delivery technologies that made the request.* (Abstract, Fig. 2 and col. 4 lines 21-64)

Regarding claim 56, which is dependent upon claim 51, Burkett further discloses:

> *wherein converting the request comprises translating the request into a document object model document having a predefined name that is included in the request and a plurality of tags having attributes indicative of a corresponding datatype.* (col. 4 lines 21-64)

Regarding claim 57, which is dependent upon claim 56, Burkett further discloses:

> *translating the data responsive to the request into another document object model document to represent an output message with datatypes that are limited to the group of predefined datatypes, and converting the another document object model into a format indicated by the request to be compatible with the one of the delivery technologies that made the request.* (Abstract and col. 4 lines 21-64)

Regarding claim 58, which is dependent upon claim 51, Burkett further discloses:

wherein limiting the datatype comprises limiting the data to representation as one of integer, Boolean, string and group. (col. 15 lines 4-67)

Regarding claim 59, which is dependent upon claim 51, Burkett further discloses:

generating a structure for a response to the request in XML that includes the data responsive to the request, wherein in the response, the data responsive to the request is limited to the predefined group of datatypes. (col. 15 lines 4-67)

Regarding claim 60, which is dependent upon claim 51, Burkett further discloses:

wherein accessing data ,responsive to the request comprises limiting the data responsive to the request that is retrieved to representation as one of integer, long, Boolean, string and group. (col. 15 lines 4-67)

Regarding claim 61, which is dependent upon claim 51, Burkett further discloses:

converting the data responsive to the request to a plurality of fields based on a datatype of the data responsive to the request; (Abstract and col. 15 line 4 – col. 16 line 4)

limiting the datatype of the data responsive to the request included in the fields to one of a predefined group of datatypes; (col. 15 line 4 – col. 16 line 4) *and*

providing the data responsive to the request as a response. (col. 4 lines 16-24)

Regarding claim 62, which is dependent upon claim 51, Burkett further discloses:

wherein extracting the request parameters comprises executing custom application code that is responsive to a request name included in the request. (col. 4 lines 16-24)

Regarding claim 63, which is dependent upon claim 62, Burkett further discloses:

> *wherein executing custom application code comprises setting the root element to a message name as a function of the request name parameter.* (col. 4 lines 16-24)

Response to Arguments

10. Applicant's arguments have been fully considered but they are not persuasive. It is noted that the amendment substantially changes the scope of the claimed subject matter.

Regarding claims 21-22, 24-30, 32-33 and 71-75, Applicant argues on pages 14-16 that the cited references are deficient because they do not teach limiting the structure of the first document object model to representation as an input message with a plurality of fields.

The Office respectfully disagrees. The Burkett reference discusses well-known DOM processing in the Background of the Invention section. In this section Burkett discloses that a DOM was tailored or modeled on the input document data. Therefore, the DOM data structure, i.e. tree, was perforce limited to the contents of the input data, as that was what the DOM was modeling. Further it was well-known to one of ordinary skill in the art at the time of the invention that the DOM tree structure has nodes corresponding to XML tags (i.e., document fields), and that XML documents and their representations are often referred to as "messages".

The Office notes that Burkett's teaching of further dynamic updates to the DOM is reflective of Burkett's ability to process a tag that indicates that certain data may be updated. Therefore, the field associated with the tag is in fact, limiting (i.e., the field is limited to a "dynamic capability". The Office also notes that Burkett's subsequent processing (i.e., dynamic

updates after an initially created DOM) merely discloses Burkett's additional capabilities. The Office further notes that the additional processing (i.e., the ability to perform dynamic updates by associating a DOM data structure field with an XML document tag indicating that data may be dynamically updated) is not germane to the issue at hand. Burkett teaches the claimed limitation, then goes on to perform subsequent activities/processing (i.e., dynamic data updates). The Office also notes that the subsequent dynamic processing performed by Burkett is enabled by the DOM data structure, which is created as having a field (i.e., a tree node) that is limited to the dynamically processed datatype.

Regarding claims 41-50, 76-77 and 79, Applicant argues on pages 16-17 that the cited references are deficient because they do not teach the amended claim language limitation reciting a MESSAGEDEFINITION class that includes a listing of pre-specified fields each of which describe a corresponding pre-specified data type, and custom application code that includes a pre-specified data type to limit the format of those fields included in an input message that do not correspond to the listing of pre-specified fields.

The Office respectfully disagrees and asserts that the amended claim language is taught in the cited passages set forth above in the rejection of the claims under 35 USC §103(a). The rationale therefore is also found above

Regarding claims 51-63, Applicant argues on pages 17-18 that the cited references are deficient because they do not teach converting a request to a plurality of fields based on request

parameters included in the request limiting a datatype of data included in the fields to a predefined group of datatypes, and extracting the request parameters based on the datatype.

The Office respectfully disagrees. The Burkett reference discusses well-known DOM processing in the Background of the Invention section. In this section Burkett discloses that a DOM was tailored or modeled on the input document data. Therefore, the DOM data structure, i.e. tree, was perforce limited to the contents of the input data, as that was what the DOM was modeling. Further it was well-known to one of ordinary skill in the art at the time of the invention that the DOM tree structure has nodes corresponding to XML tags (i.e., document fields), and that XML documents and their representations are often referred to as "messages".

The Office notes that Burkett's teaching of further dynamic updates to the DOM is reflective of Burkett's ability to process a tag that indicates that certain data may be updated. Therefore, the field associated with the tag is in fact, limiting (i.e., the field is limited to a "dynamic capability". The Office also notes that Burkett's subsequent processing (i.e., dynamic updates after an initially created DOM) merely discloses Burkett's additional capabilities. The Office further notes that the additional processing (i.e., the ability to perform dynamic updates by associating a DOM data structure field with an XML document tag indicating that data may be dynamically updated) is not germane to the issue at hand. Burkett teaches the claimed limitation, then goes on to perform subsequent activities/processing (i.e., dynamic data updates). The Office also notes that the subsequent dynamic processing performed by Burkett is enabled by the DOM data structure, which is created as having a field (i.e., a tree node) that is limited to the dynamically processed datatype.

Regarding claims 64-70, Applicant argues on pages 18-19 that the cited references are deficient because they do not teach: 1) a listing of datatypes; 2) limiting data structure fields in a DOM document; and, 3) the use of a server.

The Office respectfully disagrees.

Regarding the listing of datatypes, the Office notes that in order to construct a DOM (document object model) modeling an XML document, it was implicit that a listing of datatypes was obtained via the inherent parsing and subsequent tree structure construction performed via DOM processing.

Regarding the limiting data structure fields in a DOM document, the Office notes that this assertion was previously addressed above. The Burkett reference discusses well-known DOM processing in the Background of the Invention section. In this section Burkett discloses that a DOM was tailored or modeled on the input document data. Therefore, the DOM data structure, i.e. tree, was perforce limited to the contents of the input data, as that was what the DOM was modeling. Further it was well-known to one of ordinary skill in the art at the time of the invention that the DOM tree structure has nodes corresponding to XML tags (i.e., document fields), and that XML documents and their representations are often referred to as "messages". The Office notes that Burkett's teaching of further dynamic updates to the DOM is reflective of Burkett's ability to process a tag that indicates that certain data may be updated. Therefore, the field associated with the tag is in fact, limiting (i.e., the field is limited to a "dynamic capability". The Office also notes that Burkett's subsequent processing (i.e., dynamic updates after an initially created DOM) merely discloses Burkett's additional capabilities. The Office further notes that the additional processing (i.e., the ability to perform dynamic updates by associating a

DOM data structure field with an XML document tag indicating that data may be dynamically updated) is not germane to the issue at hand. Burkett teaches the claimed limitation, then goes on to perform subsequent activities/processing (i.e., dynamic data updates). The Office also notes that the subsequent dynamic processing performed by Burkett is enabled by the DOM data structure, which is created as having a field (i.e., a tree node) that is limited to the dynamically processed datatype.

Regarding the use of a server, the Office notes that the employment of a server in a processing architecture was well-known by one of ordinary skill in the art at the time of the invention, as evidenced, for example, in col. 6 lines 7-12 of Burkett discussing the use of servers in prior art networks. Thus Burkett does not teach away from the use of servers. It is further noted that the Applicant's claimed use of a server is merely to provide a platform for the execution of software. The cited references do not teach away from the use of a computer platform to execute software, it being noted that the selection of hardware was merely an obvious variant to one of ordinary skill in the art at the time of the invention.

For these reasons, the Office asserts the rejections under 35 USC §103(a) as set forth above.

Conclusion

11. The prior art made of record and not relied upon is considered pertinent to applicant's

US Patents

Claussen et al	6,718,516
Wanderski et al	6,519,617

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert Stevens whose telephone number is (571) 272-4102. The examiner can normally be reached on M-F 6:00 - 2:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Heather R. Herndon can be reached on (571) 272-4136. The current fax phone number for the organization where this application or proceeding is assigned is 703-872-9306. Additionally, the main number for Technology Center 2100 is (571) 272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Robert Stevens
Art Unit 2176
Date: May 25, 2006

rs

William L. Bashore
WILLIAM BASHORE
PRIMARY EXAMINER
5/29/2006